LLNL-JRNL-660558

Concept to Reality: What it Takes to Develop Your Own Database Application

D. Larson, R. Kamerzell, J. Branum

September 18, 2014

The Synergist

CONCEPT TO REALITY

WHAT IT TAKES TO DEVELOP YOUR OWN DATABASE APPLICATION

By: Diana L. Larson, Ryan G. Kamerzell, James D. Branum

When commercial off-the-shelf (COTS) data management systems do not meet the needs of your organization, methods for managing, retaining, and retrieving your Industrial Hygienist (IH) data can be called into question. A well-integrated data management system is particularly important if the database is being relied upon to be the official repository of your IH records and drive your internal processes. In an organization where programming and funding resources are available and a COTS system is not a good fit, consider developing your own database application.

Before venturing down this path, there are two key questions that need to be answered. The first question is why didn't the COTS system meet your needs? It is important to be very honest and clear about what worked, what didn't and why. The second question is what are the data management or process related issues that are vitally important to the success of your organization that the COTS system did not address or meet?

It is important to clearly identify the data that needs to be managed and the fundamental functionality of the database application. Can the COTS system simulate your existing processes? Do you need a database that is relational to one or more external data sources and is this even feasible with a COTS system? Or are there unresolved issues related to the user interface such as not being able to define organization-specific input fields? Are there issues related to functionality such as not having the ability to define or customize data outputs including reporting features that fit your specific needs? Does some information in the database need to be accessible to non-IHs in your organization and the COTS system doesn't allow you to define custom privilege sets? If the users of the COTS system have issues with data input, it will not be well received or utilized, which will result in generating records with poor data quality. When there are unresolved issues with data retrieval, data integrity or validation, or not having your data protected and accessible in a format that meets your needs, it can be worse than storing your records in a file cabinet or simply logging details on a spreadsheet.

The amount of effort required to develop your own database application can be substantial. It takes considerable forethought and an investment in time and resources. These need to be weighed against the effort and cost associated with working with a commercial provider to customize a COTS system, especially if the COTS system actually simulates your existing work processes. If the decision is made to develop your own database here are a few things that worked well for us.

Throughout the entire process from planning to production, involve and engage the end users, programmers, and your identified support staff. It is crucial that the application not be developed in a vacuum. Identify the breadth of who will be interacting with the database application. Will it be just the IHs or will non-IHs need to interface at some level with the application? Identifying the complete set of end users will focus your development of the application.

Identify programming skills available for application development. Do you have a skilled programmer who can be devoted to your project? Is there someone on your staff, like an IH, who is not a programmer but can be trained and mentored particularly for user interface design? If you have someone on your staff who can assist the lead programmer in all aspects of development, they will be able to make updates and modifications well after the application is launched to both satisfy end users and keep up with changing processes. Since a significant amount of development time is spent on the back-and-forth between the programmer and the organization who wants the application, having an IH learn a basic set of programming skills can save an immense amount of development time. Combining an IH with a programmer allowed us to make real-time decisions about process flow and data input variables from conception to production.

Identify your support staff. They are the ones who will maintain the servers where your database application resides and ensure that your data is backed up on a regular basis. They can also help identify where the program will reside and any infrastructure support that will be needed. They may also be the ones who manage your other institutional databases and can help with any needed connectivity to make your database application relational. Establish business rules with your support staff to ensure that the operation of the system is maintained. For example, who will be running the backups, who will install updates to the software or patch the operating system on your server? Develop user agreements and ensure that they are in place before you go into production.

A key decision will be to determine the software for developing your database. Several factors play into this decision: (1) What method will you use to determine the functional requirements of the database application? Will you be doing rapid prototyping or will a requirements document be written first? If you plan to do rapid prototyping for quick deployment and proof of concept, the requirements document can be written as development progresses but typically this is done first before development begins. Some software applications are better for rapid prototyping than others, so it is important to select software that lends itself to ease of use and has the desired feature sets. (2) The scope of the project. What does the database need to do and what type of data does it need to contain? Even if the database is being developed using rapid prototyping software, this step must be done first; (3) The number, job classification, and method(s) by which users will need to connect to the database. This will be one of the largest considerations when determining the final software program. For example, will this database be used by one IH or by 1000 people with varying job classifications (IH's, technicians, management, etc.)? Will some users or classes of users need to connect from mobile devices or via web browsers as well as from computers connected directly to the company network? Will the database need to support both Microsoft Windows and Apple operating systems? Some commercial products only allows up to 20 individuals to simultaneous access a database while supporting one type of operating systems. Other software allows many more simultaneous users under different operating system and may connect via the web or from Smartphones and tablets. (4) The skill level of the programmers available to develop your database system. A senior programmer who is devoted to your project will be needed if more complex programming software is selected. If the plan is to do rapid prototyping or if your programmer resource is less experienced and is working with a senior programmer, then something like FileMaker Pro or Access may be a good choice. In our case, we selected FileMaker Pro, which allowed an IH who was a less experienced programmer to develop content and prototype the processes. FileMaker Pro also met our requirements for a database that would support 50 to 100 simultaneous Windows and Apple operating system users as well as those using mobile devices and web browsers to both collect and enter data as well as to view live reports. We were able to develop and prove our processes in FileMaker Pro and avoid trial and error in more complex and time intensive software before considering converting our database to Oracle which is the company-standard for production databases. When using the rapid prototyping model for database development, be considerate to the form and function of your desired software endpoint. Special attention is needed when defining fields and naming layouts in FileMaker or Access so that once converted, it will behave the same in Oracle. Lastly you need to evaluate budget constraints for both software procurement costs and development time. Databases developed with products such as Oracle can be very robust but will take more time to program and require a higher programmer skill level. Talk to either your IT staff or a consultant before making a decision.

Once your key people are assembled, start by identifying the end products and the overall architecture of the database. What is it you need the database to do and what are the outputs. Start with the end product and work backward to identify the input variables. The breadth of the data that is needed for specific processes becomes much clearer in this case. You can then identify if your database needs to interface with any of your other database systems. If so, engage your IT support folks or a consultant to identify what connections are needed for providing live data feeds. They can help determine how the data will be fed and how it will be refreshed. If the programming languages between the databases are different, then what tools are needed for setting up the data extraction? All too often you can be caught up in populating front-end tables and generating lists or algorithms that you *hope* will be useful in developing the outputs and end up not having your needed outputs.

When designing the database application, maintain vigilance for compliance with internal and external requirements. Controls on data access for specific users and/or roles of users, security of transmitted and stored data, and data retention periods are among those subject areas frequently addressed by contracts, regulations, and statutes, in addition to your own company's policies and practices. If your company is OHSAS 18001 or ISO 9001 certified, design your database system with data quality and continual improvement in mind. For example, if you have a survey that is reoccurring, like periodic surveys of a workplace, design the database to automatically capture and append issues identified from the previous survey to the current survey record so you can assess and determine the effectiveness of any corrective actions that were taken in the past. Data history and revision logging are essential for maintaining data integrity and are essential if the database will be used to house your official records. However, programming this layer into the database will take additional time to develop.

When designing the user interface screens for data entry, consider building in as many tools and including as much logic as possible to avoid common data entry errors. Also put blocks on navigation so data isn't missing that is required for a complete record. Developing tools that require specific data input fields for running an algorithm helps minimize these types of data quality issues. If specific fields must be completed, add logic so that the screen cannot be saved or that the next data field cannot be entered before completing a required entry. We have found that color coding fields adds additional visual cues to a user so that these fields stand out as being required.

Once these decisions have been made, it is time to develop the project plan. Identify the user interfaces from the various data entry points all the way to output reporting and how each of them will function. These plans can be a simple flowchart or an elaborate document, depending upon the needs and complexity of your processes within your organization, including other external data requirements. Often existing written procedures will outline your database application requirements. This is basically the plan of who does what and how data flows. More elaborate plans may include timelines with completion dates for key deliverables.

During the requirements development phase, consider your organizational requirements for accessing the database. How will end users gain access and what is needed? Based on the platform they are using, what computer security or data protections are needed? It is essential to work closely with your programmer when developing these documents as this will be the roadmap, or in some cases the contract, for the database application development. This is how your developer will know when this project is completed.

After the plan is finalized, identify what resources are needed to support the database application development and eventual production launch. This also includes maintenance and day-to-day operation of the database system. Estimate the cost and determine if the necessary resources are currently available or if budget/resource constraints need to be resolved before development or production can begin.

As code development begins, continuously engage the end users and hold regular meetings to solicit their feedback. Identify at what point beta testing can begin. Often as soon as a function module or a core feature is completed, this is a good time to ask the beta testers to provide honest feedback on the functionality and user interface screens. It is important to address their feedback in a timely manner so that they remain engaged in the process. When the development is completed, these are the folks who will be your biggest advocates and will help socialize the database application to the other users. Keep all end users informed on the development process, letting them know what to expect and when. Communication is key to the acceptance of evolving processes.

During the beta testing phase, procedures for the end users need to be written. This will add consistency for data inputs when multiple users interact with the same database system. In some cases, when the database output interfaces with different groups within your organization, business rules may need to be developed. For example, if a report recommending specific actions is sent to a different work group outside of your organization, they need to know ahead of time when these reports may be generated and what types of actions they may need to address. Business rules help clarify everyone's roles and responsibilities for managing database outputs.

It will be a decision among your working group to determine when the database application system is ready for production. Hold meetings with the end users during the roll-out phase and review any procedures associated with the use of the database. It is helpful to have some of your beta testers in the audience to help answer questions. Change always has a learning curve and may be met with opposition from workers who have longevity with your organization or those that may not have strong computer skills. Keep in mind that your database system is a work in progress and changes in technology can have significant impacts to your database. Anticipate the need for version updates, not only to the main database software, but also to any other software that may be used with it for linking to other data sources, including third party applications, both internal and external to your organization.

Once the database is launched, ensure that programming support is available to help resolve any bugs, make updates and changes to database interfaces. This is the care and feeding to ensure that the database continues to function the way it was initially designed and launched. This support is essential in the first few weeks after production. Plan accordingly to make sure any bugs can be fixed in near real-time during these first few weeks.

Allow time for the new processes to begin and users to adapt to the start-up of the database application before taking time to reflect on what was learned. Then ask yourself the hard question, did we make the right choice and was it worth it? What benefits did we achieve by developing our own system? For the development of our database application, there was no doubt it was worth every bit of effort including all the challenges. One of the greatest confirmations we have received is that each time we have shared our work with other organizations, the feedback has been incredibly positive. In the end, this is not only a measure of success in having a functioning database that uniquely meets our needs but personal satisfaction as well.